IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

First Named Applicant: Preisig	)	Art Unit: 2177
Serial No.: 09/871,475	)	Examiner: Le
Filed: May 31, 2001	)	**SVL9-2001-0020-US1**
For: **SYSTEM, METHOD, AND COMPUTER PROGRAM PRODUCT FOR REFORMATTING NON-XML DATA FOR USE WITH INTERNET BASED SYSTEMS**	)	February 24, 2004 750 B STREET, Suite 3120 San Diego, CA 92101

RULE 132 DECLARATION

Commissioner of Patents and Trademarks
Washington, DC 20231

Dear Sir:

I, Arthur Ryman, declare as follows:

I am one of the original inventors of the invention claimed in the above-captioned application.

As evidenced by the enclosed paper dated May 16, 2000 entitled "Lunar Eclipse Proposal: Opportunistic B2B Tools" listing me as author and by the slide on the Lunar Eclipse project which I presented on June 9, 2000, I conceived of the present invention at least prior to June 26, 2000. Specifically, using the limitations of Claim 1 as an example and referring to the enclosed documents, a client is shown on the slide as is a server (both the DB2 server and relational database (labeled "RDB" in the slide). Also, middleware is shown communicating with the client and server. This middleware is the large box in the slide that includes, among other things, a "servlet", a "HTTP" section, and a "SOAP" section.

As set forth in page 3 of the enclosed paper, the middleware includes at least one file. Specifically, under the heading "HTTP GET and POST", it is disclosed that the result of HTTP GET or POST is an HTML document, which documents are well known to be stored in files. It is further disclosed in this section that the file from either one of the GET/POST or XML processing includes parameters for service invocation.

Both the slide and page 6 of the enclosed paper clearly show that the parameters referred to above are database system query language statements. Specifically, the server shown in the slide is clearly a database, and on page 6, under "XML Enabled Relational Database" and "Developing New Database Services", it is disclosed that the preferred database is DB2. Also, under "Developing New Database Services", it is disclosed that the parameterized statement is used to return data satisfying the request to the client ("return a query result as an XML document"). The requests are from clients, see the slide and page 2 discussion in the paper of responding to client requests for flight information.

I declare that we were diligent in reducing the invention to practice at least from a time prior to June 26, 2000. Specifically, I declare that we diligently developed the Lunar Eclipse program on a daily basis from a time prior to June 26, 2000 and diligently submitted an invention disclosure to IBM Intellectual Property Department, which then diligently processed the application for disclosure to outside counsel and subsequent filing within the usual course of IBM business in filing patent applications.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United State Code and that such willful, false statements may jeopardize the validity of the application or any patent issued thereon.

Arthur Ryman

BY: ARTHUR RYMAN

date: 2004-02-23

Respectfully submitted,

John L. Rogitz
Registration No. 33,549
Attorney of Record
750 B Street, Suite 3120
San Diego, CA 92101
Telephone: (619) 338-8075

JLR:jg

FAX 619 338 8078

Lunar Eclipse Proposal: Opportunistic B2B Tools

Arthur Ryman
2000-05-16

Introduction

XML as an Information Interchange Format

The Web is currently undergoing a rapid evolution from being a medium for, primarily, information presentation to one that also supports true information interchange. Just as Hypertext Markup Language (HTML) became the standard for information presentation, Extensible Markup Language (XML) is becoming the standard for information interchange. Although an HTML document can contain imbedded information that can be further processed in a Web browser by scripts, most of the content is intended for human readers. In contrast, XML provides a format for packaging information in a self-describing way that is suitable for further processing by programs.

The Emergence of B2B Applications

Applications that are now targeted at Web browsers (business-to-consumer, or B2C) will be repurposed to allow interaction with both more sophisticated clients as well as autonomous programs (business-to-business, or B2B), and entirely new applications will be developed. Some of these B2B applications will have modest demands for security, reliability, performance, and scalability, while others will be mission critical. Some will be ad hoc projects to quickly test new ideas while others will require deep integration with established Enterprise Information Systems (EIS).

Opportunistic versus Systematic Development

Because of the diversity of potential B2B applications, it is useful to classify them by various characteristics. Without claiming any precision, we will use the term *Opportunistic* to describe tactical applications that attempt to rapidly exploit opportunities presented by the Web, and the term *Systematic* to describe strategic applications that extend EISs to the Web. We expect that, over time, successful Opportunistic applications will evolve into Systematic applications.

Lunar and Solar Eclipse

This proposal describes *Lunar Eclipse* (aka VA/2000 for B2B, Professional Edition), a tool suite based on the Eclipse platform (aka VA/Base) that is aimed at the Opportunistic B2B development market. *Solar Eclipse* (aka VA/2000 for B2B, Enterprise Edition), the corresponding product for the Systematic B2B development market, will be described in another proposal.

The Transition from B2C to B2B Applications

Just as manual office procedures provided the models for the first computer applications, it is reasonable to expect that existing B2C applications will provide the models for the first B2B applications. Businesses will continue to attempt to improve efficiencies and reduce operating costs by eliminating

human intervention wherever possible, and when this is accomplished, entirely new applications will become possible. The next wave of new applications will be enabled by companies providing B2B *services* instead of B2C applications.

A B2B Travel Planning Application

As an example, consider the task of making travel arrangements. Private individuals can now bypass human travel agents by making travel plans over the Web. For example, the Air Canada Web site allows travelers to search for suitable flights and make reservations for them. Many Web travel brokers that aggregate multiple airlines also exist. However, an employee of a corporation may not be able to use these applications because corporate policy might require that they be made through a contracted travel agency. Nevertheless, airlines could begin to evolve their B2C applications into B2B services that interoperated better with corporate procedures, and eventually completely disintermediate travel agencies.

For example, within IBM Canada, a Lotus Notes travel planning application is used. This application allows the user to launch a Web browser on the Air Canada Web site, and then manually transcribe the flight information into the travel form. As a first step towards better integration, the Air Canada Web site could provide a B2B service that allowed other applications to query it for suitable flights and obtain the result in XML format. The Lotus Notes travel application could make such a query on behalf of the user, display the flights, and allow the user to make a select the desired flight without the need for manual transcription. Taking this a step further, the airline could also provide a reservation service. The travel application could apply business rules to verify that the flight met corporate guidelines, and then send a digitally signed reservation request to the airline. The service would respond by confirming the reservation and e-mailing the e-ticket to the user. If hotels and car rental companies provided similar B2B services, then need for human travel agents would virtually disappear.

Further cost savings would accrue from Web travel brokers, or e-marketplaces, that aggregated the services of multiple airlines, hotels, and car rental companies, and made them available to many travelers. Instead of querying a single preferred airline, the corporate travel application would query a Web travel broker that broadcast the request to many providers.

The cost efficiencies that could result from a well-designed Web travel broker might far exceed anything that human travel agents could achieve. For example, travelers with flexibility in their schedules could alter their plans to take advantage of cheaper rates, while those with tight requirements could bid on scarce seats. Travelers that owned highly sought-after tickets, but were prepared to fly at alternate times, could put them up for auction and use the profits to further defray their expenses. As Web access becomes more pervasive through the use of devices like Wireless Access Protocol (WAP) cell phones, travelers could remain involved in these transactions up to the moment they stepped on the plane. And, of course, if your flight plans changed, the Web travel broker would automatically update your hotel and car reservations.

B2B Services

A B2B service is a set of related application functions that are made accessible to other programs over the Internet and are designed for information interchange as opposed to information presentation. By developing and hosting B2B services, companies can gain competitive advantage through access to new sales channels and by enabling novel e-marketplace applications.

Transport Protocols

B2B services can be delivered by many transports, but because of its pervasiveness and ability to pass through corporate firewalls, Hypertext Transfer Protocol (HTTP), with optional Secure Sockets Layer (SSL), i.e. HTTPS, will probably become the dominant one. Other likely important transports include Simple Mail Transfer Protocol (SMTP) and File Transport Protocol (FTP). The proprietary IBM MQ messaging protocol could gain acceptance because of its dominant market share, but historically the Internet has favoured open standards. The distributed object protocols Java Remote Method Invocation (RMI), CORBA Internet InterORB Protocol (IIOP), and Microsoft Distributed COM (DCOM) will probably be confined to communication within an enterprise because they are complex and strongly couple the implementation technologies used by the connection endpoints. In contrast, HTTP is very neutral about the choice of service implementation technology and is therefore better suited to the highly heterogeneous Internet.

HTTP GET and POST

HTTP includes rudimentary support for B2B service invocations through the GET and POST request methods. Both GET and POST were designed to send simple name-value string pairs, typically collected through an HTML form, as parameters for the service invocation. The result of an HTTP GET or POST is normally an HTML document. Although this simple scheme is adequate for B2C applications, it is unwieldy for B2B services. The invocation parameters for a B2B service may be complex and the result is normally subject to further processing by programs. XML provides a way to send complex parameters and return results in a readily processed format, and several remote procedure call standards based on XML have been proposed.

SOAP

IBM has recently cooperated with Microsoft on the definition of the Simple Object Access Protocol (SOAP) as a way to invoke B2B services. SOAP defines an XML format for sending parameters to and receiving results from B2B services. SOAP works well with HTTP, but can also be used with other transports. SOAP provides a key part of the infrastructure needed for B2B services, but more is needed to complete the picture. These include a service description language, a way to publish service descriptions, and bindings for key development languages such as Java.

Describing Services

In order for an application to use a B2B service, the service interface must be precisely described. The service description language plays the same role as Interface Description Language (IDL) for CORBA or COM. The service description language provides clear documentation to application developers, and is used by tools that generate code (client stubs or connectors) that hides protocol details and allows the application to invoke the service.

Publishing Service Descriptions

A company's repository of service descriptions must be made easily accessible to application developers. One possible way to publish service descriptions is through the use of the Distributed Authoring and Versioning (WebDAV) Protocol. WebDAV extends HTTP 1.1 with the addition of collections, properties, and resource locking. A WebDAV property is a name-value pair where the name is a Universal Resource Identifier (URI) and the value is string or XML element. A standard set of property names, analogous to the Dublin Core for document indexing, could be defined for services, and servers could then be queried using the WebDAV PROPFIND method for the services that they provide.

Companies that provided generic services could register with specific service search engines or make their repositories generally accessible to service Web crawlers. If a service from a specific company was being used, the developer could directly access its WebDAV service repository, but if the developer was looking for a generic service, then a service search engine or Web crawler could be used to locate a suitable provider.

Java Bindings

Since the Internet is very heterogeneous, service descriptions must be neutral to the technologies in which the service is implemented if they are to become widely accepted by the development community. In particular, developers must be able to implement service clients in the technology of their choice. Developers must bind their chosen implementation technology to the service description.

However, within an implementation technology, such as Java, standard bindings are useful in order to promote portability and create a viable market for Independent Software Vendors (ISV). For example, the developer of a Java service should be able to deploy it on servers from different vendors with no change to the source code of the service. This goal has been achieved for Enterprise JavaBeans (EJB) through the use of deployment descriptors. Standard XML deployment descriptors for Java services should also be defined.

Target Runtime

The goal for Lunar Eclipse is to appeal to opportunistic developers who are just beginning to explore the possibilities of B2B applications. It is therefore important to get to market quickly with a simple offering that nevertheless allows the development of useful applications. The target runtime for these applications must therefore also be simple, with the requirements of the more complex systematic developers being addressed by Solar Eclipse. The proposed runtime consists of:

- Web server
- servlet/JSP engine
- XML parser
- XSLT processor
- Services module
- SOAP module
- WebDAV module
- XML enabled relational database

The target runtime operating systems, in order of importance, are probably:

- Windows NT/2000
- Linux
- AIX

The following runtime components are not directly supported by Lunar Eclipse, but neither is their use excluded:

- SMTP server
- FTP server
- EJB server
- message queue

- workflow engine
- other Enterprise Application Integration (EAI) middleware

Web Server

The preferred Web servers are the IBM HTTP Server and Apache, but any Web server that supports a compatible servlet/JSP engine can be used.

The Web server provides support for HTTP as well as authentication and access control. It is assumed that B2C and B2B applications will coexist on the same server.

Servlet/JSP Engine

The preferred servlet/JSP engines are WebSphere Standard Edition and Apache Tomcat, but any compatible servlet/JSP engine can be used. The support for Apache Tomcat is critical to winning mindshare in the Java development community.

It is assumed that Java services or their proxies will run in the servlet engine. JSPs can be used for generating HTML for presentation in B2C applications, as well as for generating XML for information interchange in B2B applications.

XML Parser

The preferred XML parser is IBM XML4J which is available as the Apache Xerces project.

The XML parser is used by the Java services, servlet engine, SOAP module, and possibly other runtime components.

XSLT Processor

The preferred XSLT processor is LotusXSL which is available as the Apache Xalan project.

Java services will use XSLT to transform XML results generated by queries or other services.

SOAP Module

The preferred SOAP module is IBM SOAP4J which was recently released on alphaWorks. However, for this implementation to become widely accepted it probably has to become part of the Apache Jakarta project and released with Tomcat. Plans are underway to make this happen.

There is a competing SOAP implementation by SOAP contributor Don Box of Developmentor. It may also be necessary to support this implementation in order to appeal to the existing SOAP developer community.

The SOAP module runs in the servlet engine and route SOAP requests to Java services. The SOAP module unmarshalls the parameters in the request, invokes the correct service, and marshalls the result into the response.

Although we refer to it as the SOAP module, it should also support services that are invocable by HTTP

GET and POST.

WebDAV Module

The preferred WebDAV module is Apache mod_dav. The earlier IBM DAV4J implementation is deprecated. Any compliant WebDAV module can be used.

The WebDAV module will be used to publish service descriptions.

XML Enabled Relational Database

The preferred database is DB2 with the XML Extender. Since there is a lack of standards for XML database access, other databases will only be supported through JDBC drivers with corresponding lack of function.

A relational database is an essential component of most Web applications. The ability to store XML in databases and to return query results as XML greatly simplifies the development of B2B applications.

Development Tasks

Lunar Eclipse is intended to support both B2C and B2B development. B2C development tasks are well-understood and will not be described in detail here. They include HTML, JSP, servlet, JavaBean, and database access development. The support for B2C development tasks in Lunar Eclipse will be minimal in comparison with WebSphere Studio and VisualAge for Java. Instead, Lunar Eclipse will focus on B2B develop tasks.

The main B2B development tasks supported by Lunar Eclipse are:

- develop new database services
- develop new Java services
- invoke existing services from Java

Developing New Database Services

Many useful B2B services can be implemented as relational database queries and updates, and require no Java programming. Services that require more complex business logic can be implemented as stored procedures, or as Java services that perform database access.

The DB2 XML Extender provides the following functions:

- store an XML document in a column
- store an XML document in a set of tables as relational data
- return a query result as an XML document

The developer should be able to define a database operation using the DB2 XML Extender and deploy it as a B2B service.

Developing New Java Services

B2B services that require complex business logic can be implemented in Java. The developer should be able to develop a Java class that implements the business logic of the service and deploy it as a B2B service without having to implement any protocol details. At most, the Java developer may have to create a deployment descriptor for the service. Tools would then automatically generate any necessary protocol handling code from the deployment descriptor.

Although Lunar Eclipse does not support the development of EJBs or command beans that access EISs, it does support making these beans accessible as B2B services.

Invoking Existing Services from Java

A Java component may need to access other existing B2B services. The developer should be able to obtain a service description through a variety of means, and then automatically generate a Java connector to the service. The connector would render the service interface as a proxy that hides all of the protocol details.

Development Tools

The target operating systems for Lunar Eclipse are, in probable order of importance:

- Windows 2000/NT/9x
- Linux

Although Windows 9x is not a target runtime platform, developers must be able to install the runtime on Windows 9x for debugging and testing. This requirement can be solved by using Apache Tomcat.

Eclipse

All development tools will be integrated into the Eclipse platform which provides common services such as the project model, team support, and user interface.

Web Tools

The Web tools support HTML and JSP editing, database access, and site management. These tools are currently being migrated from WebSphere Studio to Eclipse.

Java Tools

The Java tools provide a simple Integrated Development Environment (IDE) including source browsing, editing, compilation, and debugging. These tools are currently being migrated from VisualAge for Java to Eclipse.

XML Tools

The XML tools support the development of DTD, XML Schema, XSL, and other XML technologies. They also support database access, but this function needs to be aligned with DB2 XML Extender. These tools are being migrated from the Visual XML Tools released on alphaWorks to Eclipse.

B2B Services Tools

The B2B services tools suport the deployment of database and Java services, and the access of B2B services from Java. These tools do not yet exist but should be based on the Enterprise Access Builder framework used in VisualAge for Java.

Conclusion

The Web development community is poised for an explosion of B2B applications. Although the B2B infrastructure may take several years to mature, it is critical to enter the market now in order to win mindshare. IBM can successfully enter this market by providing a simple but useful runtime and tool offering that is based on Java and XML standards and Open Source implementations.

The benefit of winning in the low end of B2B applications is that IBM establishes mindshare and positions itself well to upsell developers when they graduate to distributed object and message based middleware. Failure to do so could result in the complete domination of the low end by Microsoft and lock developers into Microsoft DNA technology as their middleware needs mature.

e-business

Lunar Runtime